WHAT IS CLAIMED IS:

1. A block matching processor for flexibly supporting block matching motion estimation at motion vector prediction modes for a plurality of matching blocks of pixels having non-uniform sizes that are multiples of a smallest matching block of said plurality of matching blocks, said processor comprising:

a plurality of difference unit (D-unit) arrays for generating an absolute value of each smallest size matching block in each D-unit array, and said each D-unit array comprising a plurality D-units arranged to correspond with an arrangement of pixels of said each smallest size matching block in each D-unit array for calculating the differences between the pixels of a current frame and the pixels of a reference frame, and converting the differences to absolute values; and

an accumulator connected to the D-unit arrays for generating a SAD (Sum of Absolute Difference) for said each smallest size matching block and a SAD for all of the plurality of matching blocks of pixels having non-uniform sizes by hierarchical addition of the absolute values of the smallest size matching blocks received from the D-unit arrays.

2. The block matching processor according to claim 1, wherein the accumulator comprises a two-stage unit forming a binary-tree structure for hierarchical addition.

3. The block matching processor of claim 1, further comprising a comparator unit connected to the accumulator for selecting one of the SADs of the matching blocks non-uniform sizes of according to a motion vector prediction mode, for comparing the selected SAD with the previous stored SAD, for selecting a smaller SAD of the two, and for storing the smaller SAD.

4. The block matching processor according to claim 1, wherein the smallest size of said matching blocks having non-uniform sizes is 4x4.

5. The block matching processor according to claim 1, wherein the smallest size of said matching blocks having non-uniform sizes is non-symmetrical.

6. The block matching processor according to claim 5, wherein the smallest size of said matching blocks having non-uniform sizes is 4x8.

7. A block matching processor for flexibly supporting block matching motion estimation at motion vector prediction modes, where a size of matching blocks used is one of 8×8, 16×8 and 16×16 pixel sizes, comprising:

a plurality of difference unit (D-unit) arrays for generating absolute values of each 8×8 matching block, each D-unit array having D-units arranged corresponding to the pixels of said each 8×8 matching block for calculating the difference between the pixels of a current frame and the pixels of a reference frame, and converting the differences to absolute values; and

an accumulator connected to the D-unit arrays, for generating a SAD (Sum of Absolute Difference) for said each 8×8 matching block and a SAD for a 16×8 matching block by hierarchical addition of the absolute values of the 8×8 matching blocks generated by and received from the plurality of D-unit arrays.

8. The block matching processor according to claim 7, wherein the accumulator comprises a two-stage unit forming a binary-tree structure for hierarchical addition.

9. The block matching processor of claim 7, wherein the accumulator comprises:

a pair of first level accumulators connected to each D-unit array, each first level accumulator generating a SAD for one of two 8×4 sub-blocks separated from an 8×8 matching block by adding absolute values of the sub-block based on the absolute values of the 8×8 matching block received from a D-unit array connected to the first level accumulator; and

a pair of second level accumulators, each second level accumulator connected to two pairs of first level accumulators, for generating a SAD for the 8×8 matching block by adding the SADs of two sub-blocks received from the connected first level accumulators and generating a SAD for a 16×8 matching block by adding the SADs of the 8×8 matching blocks.

10. The block matching processor of claim 9, wherein each D-unit calculates the difference between pixels of said current frame and pixels of said reference frame using a ripple-carry adder (RCA).

11. The block matching processor of claim 9, wherein each first level accumulator is of a hierarchical carry-save adder (CSA) binary-tree structure with carry-save (4, 2) counters connected in a quad-binary-tree configuration.

12. The block matching processor of claim 10, wherein each first level accumulator is of a hierarchical carry-save adder (CSA) binary-tree structure with carry-save (4, 2) counters connected in a quad-binary-tree configuration.

13. The block matching processor of claim 11, wherein each second level accumulator is a binary-tree structure comprising two carry-save (4, 2) counters and two carry-propagate adder (CPA) for generating the SAD of an 8×8 matching block by adding the SADs of two sub-blocks and a third CPA for

generating the SAD of a 16×8 matching block by adding the SADs of two 8×8 matching blocks.

14. The block matching processor of claim 12, wherein each second level accumulator is a binary-tree structure comprising two carry-save (4, 2) counters and two carry-propagate adder (CPA) for generating the SAD of an 8×8 matching block by adding the SADs of two sub-blocks and a third CPA for generating the SAD of a 16×8 matching block by adding the SADs of two 8×8 matching blocks.

15. The block matching processor of claim 7, further comprising a comparator unit for generating a SAD of a 16×16 matching block by adding the SADs of two 16×8 matching blocks generated from the single pair of second level accumulators, selecting one of the SADs of the 8×8, 16×8, and 16×16 matching blocks according to a motion vector prediction mode, comparing the selected SAD with the previous stored SAD, selecting the smaller of the two, and storing the smaller SAD.

16. A method for flexibly supporting block matching motion estimation at motion vector prediction modes for a plurality of matching blocks of pixels having non-uniform sizes that are multiples of a smallest matching block of said plurality of matching blocks, comprising the steps of:

(a) generating an absolute value of each smallest size matching block in each D-unit array of a plurality of difference unit (D-unit) arrays, wherein said each D-unit array comprising a plurality D-units being arranged to correspond with an arrangement of pixels of said each smallest size matching block in each D-unit array;

(b) calculating differences between the pixels of a current frame and the pixels of a reference frame by each D-unit, and

(c) converting the differences calculated in step (b) to absolute values; and

(d) generating a SAD (Sum of Absolute Difference) for said each smallest size matching block and a SAD for all of the plurality of matching blocks of pixels having non-uniform sizes by hierarchically adding the absolute values of the smallest size matching blocks received from the D-unit arrays by an accumulator connected to the D-unit arrays for generating a SAD (Sum of Absolute Difference) for said each smallest size matching block, and

(e) generating a SAD for all of the plurality of matching blocks of pixels having non-uniform sizes hierarchically adding the absolute values of the smallest size matching blocks received from the D-unit arrays.

17. The method according to claim 16, wherein the hierarchical addition made by providing the accumulator with a binary-tree structure.

18. The method according to claim 16, further comprising:

(f) selecting one of the SADs of the matching blocks non-uniform sizes of according to a motion vector prediction mode; and

(g) comparing the selected SAD with a previously stored SAD to determine a smaller SAD of the two, and

(h) storing the smaller SAD determined in step (g).

19. The method according to claim 16, wherein the smallest size of each matching block in step (a) is 4x4.

20. The method according to claim 16, wherein the smallest size of each matching block in step (a) is 4x8.

21. The method according to claim 16, wherein the smallest size of

each matching block in step (a) is 8x8.

22. A method for flexibly supporting block matching motion estimation at motion vector prediction modes, where a size of matching blocks used is one of 8×8, 16×8 and 16×16 pixel sizes, said method comprising the steps of :

(a) generating absolute values of each 8×8 matching block for each D-unit array of a plurality of difference unit (D-unit) arrays having D-units arranged corresponding to the pixels of said each 8×8 matching block;

(b) calculating a difference between the pixels of a current frame and the pixels of a reference frame,

(c) converting the differences calculated in step (b) to absolute values;

(d) generating a SAD (Sum of Absolute Difference) for said each 8×8 matching block and a SAD for a 16×8 matching block by hierarchically adding the absolute values of the 8×8 matching blocks generated in step (a) by the plurality of D-unit arrays from an accumulator connected to the D-unit arrays.

23. The method according to claim 22, the hierarchical adding is performed by an accumulator having a pair of first level accumulators and a pair of second level accumulators, said pair of first level accumulators being connected to each D-unit array,

generating a SAD for one of two 8×4 sub-blocks separated from an 8×8 matching block by each first level accumulator adding absolute values of the sub-block based on the absolute values of the 8×8 matching block received from a D-unit array connected to the first level accumulators; and

generating a SAD for the 8×8 matching block by adding the SADs of two sub-blocks received from the connected first level accumulators and generating a SAD for a 16×8 matching block by adding the SADs of the 8×8 matching blocks by said pair of second level accumulators.

24. The method according to claim 22 wherein the calculating in step (b) of the difference between pixels of said current frame and pixels of said reference frame is performed by using a ripple-carry adder (RCA).

25. The method according to claim 23, wherein each first level accumulator generating the SAD for one of the two 8x4 sub-blocks is of a hierarchical carry-save adder (CSA) binary-tree structure with carry-save (4, 2) counters connected in a quad-binary-tree configuration.